AUXILIARY MIDPOINT LEARNING FOR GOAL-REACHING REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose *Auxiliary Midpoint Learning* (AML), a technique for quasi-metric learning in goal-reaching reinforcement learning (GRRL). In addition to standard temporal-difference (TD) learning of the policy and quasi-metric networks, AML jointly trains a network that predicts midpoints with respect to the learned quasi-metric and uses these predictions to improve the global consistency of the quasi-metric network. This enables more effective learning on goal-reaching tasks with long horizons. Experiments show that AML consistently improves over the quasi-metric TD baseline and achieves the highest success rate among the compared methods on several benchmark environments.

1 INTRODUCTION

Goal-Reaching Reinforcement Learning (GRRL) (Wang et al., 2023), also simply called Goal-Conditioned Reinforcement Learning (GCRL) in some works (Bortkiewicz et al., 2025), is a type of reinforcement learning (RL) where the policy is trained to reach a given goal state. Unlike standard reinforcement learning, the policy receives not only the current observation but also the goal to be reached as input. After training, it is expected that the policy can reach arbitrary given goals. This formulation is related to world model approaches (Duan et al., 2024; Mendonca et al., 2021), and could serve as a key element in realizing general-purpose AI, including physical AI.

It is an important fact that the GRRL formulation defines a quasi-metric structure on the state space (Micheli et al., 2020). Therefore, approaches with quasi-metric learning have been proposed for GRRL. The quasi-metric learning (Wang and Isola, 2022b; Liu et al., 2023) technique enables learning while satisfying quasi-metric constraints at the architecture level. Instead of traditional temporal difference (TD) learning, *Quasi-metric Reinforcement Learning* (QRL) (Wang et al., 2023) maximizes the quasi-metric distances under the constraint of adjacent states. Compared to TD learning, QRL is expected to converge faster to the optimal value function because it directly learns the quasi-metric structure. However, since it solves the constrained optimization problem with Lagrangian relaxation, it has a drawback that the learning process tends to be unstable. Moreover, in both TD learning and QRL, since long-horizon planning is difficult to learn, the predicted distances are not accurate for distant pairs (Kobanda et al., 2025). While the overestimation of distances is prevented by the quasi-metric architecture, the underestimation of distances is another problem.

Goal-reaching reinforcement learning is related to the Floyd-Warshall algorithm for the all-pairs shortest-path problem (Floyd, 1962), because the intermediate states can be used to update the value function between distant states. Floyd-Warshall reinforcement learning approaches (Kaelbling, 1993; Dhiman et al., 2018) train the value function to satisfy the triangle inequality with Floyd-Warshall-like updates, but this is unnecessary if the quasi-metric learning architecture is used. On the other hand, in the context of path planning, a method using *midpoint prediction* to learn a quasi-metric and geodesic generation from local information of quasi-metric has been proposed (Kasaura, 2025). In this method, a network to predict the midpoint between two points is trained simultaneously with a value network, and predicted midpoints, rather than arbitrary intermediate states, are used to update the value network. Theoretically, this approach also prevents the underestimation of distances, which cannot be prevented by a quasi-metric learning architecture alone. We apply this idea to the general GRRL framework and propose a technique called *Auxiliary Midpoint Learning* (AML). While learning the quasi-metric and policy locally through standard TD learning, AML simultaneously learns midpoint prediction and uses it for learning the global quasi-metric structure.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

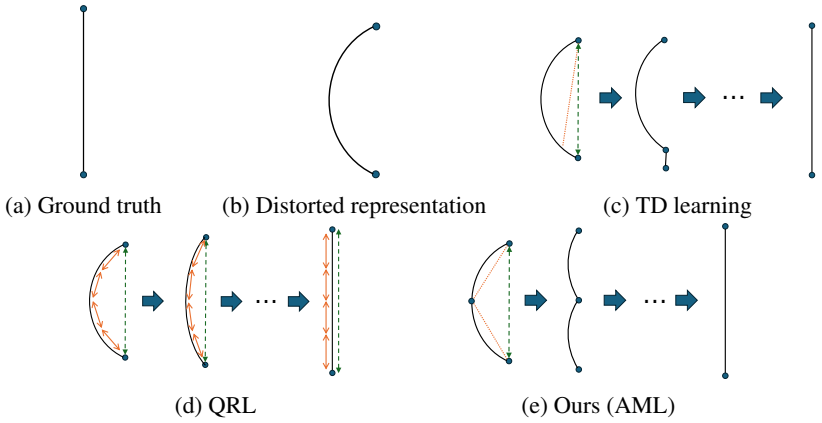


Figure 1: Figurative explanation of the concept of methods for learning quasi-metric. (a) The ground truth of the quasi-metric. (b) The current representation of the state space. Since the true geodesic is curved in the embedded space, the represented distance (straight-line distance) between two points is shorter than the true distance. (c) In TD learning, since the current distance (the dashed line) is shorter than the sum of the distance from the neighbor state to the goal (the dotted line) and that from the current state to the neighbor state, it is stretched in the next step. Iterating this process corrects the representation step by step. (d) In QRL, the global distance is directly maximized (the dashed line) under the constraint of adjacent states (the solid double-headed arrow). (e) In our proposed method (AML), the distance (the dashed line) is stretched to the sum of the distance from the current point to the midpoint and that from the midpoint to the goal (the dotted lines).

Figure 1 is a figurative explanation of the concept of our proposed method. For clarity, we consider here a situation in which the state space is embedded in the Euclidean plane. We assume that the represented distance is underestimated because the embedded geodesic is distorted. In TD learning, the distance is stretched step by step. However, because this modification is local, it takes numerous iterations to obtain a globally correct representation. On the other hand, in QRL, the global distance is directly maximized. The learning proceeds under the tension between maximization and constraints. In particular, even when the correct quasi-metric is obtained, this tension remains. In our proposed method (AML), the process in which the distance increases based on the sum of shorter distances is the same as in TD learning. However, since the intermediate state used to update the value function is not the next state but a midpoint, it can modify the representation globally in one step. Thus, AML is expected to obtain the correct representation in fewer iterations than TD learning.

Experimentally, we evaluate our method with the codebase of JaxGCRL (Bortkiewicz et al., 2025). Our method consistently improved the performance of the quasi-metric TD learning approach and achieved the highest success rate among the compared methods in several benchmark environments.

2 Related Work

Goal-Conditioned Reinforcement Learning (GCRL) (Schaul et al., 2015) trains a universal policy for various goals, which takes both the current state and the goal to be reached as input. While our GRRL framework is sometimes also called GCRL as in Bortkiewicz et al. (2025), we interpret the term 'GCRL' as a more general framework that allows the reward to depend freely on the goal and consider GRRL as a subclass of GCRL.

2.1 Representation Learning for GRRL

Micheli et al. (2020) pointed out that the value function to be learned has a quasi-metric structure. Quasi-metric learning (Wang and Isola, 2022b;a) is a technique to learn a function while satisfying quasi-metric constraints at the architecture level. Liu et al. (2023) used this technique to learn a cost function for goal-reaching reinforcement learning. Our study uses a method similar to this approach as a baseline. Wang et al. (2023) also adopted quasi-metric learning and proposed a new learning

method called *Quasi-metric Reinforcement Learning* (QRL) for GRRL, in which the quasi-metric is maximized under the constraint of adjacent states. We compared our method with QRL in the experiments.

Contrastive Reinforcement Learning (CRL) (Eysenbach et al., 2022) uses contrastive learning to represent the structure on the state space by embedding both state-action pairs and goals into a Euclidean space. It trains the embedding so that the distance between the embedding of a state-action pair on a trajectory and that of a goal to be reached on the same trajectory is small. However, the representation ability of this method seems to be limited. Indeed, metric spaces which can be embedded isometrically into Euclidean spaces are limited. While this fact does not directly apply to the case of CRL because it uses different embedding functions for state-action pairs and goals, it appears to be impossible to give a precise representation for arbitrary metric spaces. We included CRL as a compared method in the experiments.

2.2 Floyd-Warshall Reinforcement Learning

Kaelbling (1993) and Dhiman et al. (2018) pointed out that goal-reaching value functions are related to the Floyd-Warshall algorithm for the all-pairs shortest-path problem (Floyd, 1962), as these functions can be updated by finding intermediate states. They proposed methods that use brute force to search for intermediate states, which depend on the finiteness of the state spaces. The idea of using sub-goals for reinforcement learning as options was suggested by Sutton et al. (1999). Jurgenson et al. (2020) linked this notion to the Floyd-Warshall approach and proposed a method to learn the generation of intermediate states with reinforcement learning. Kasaura (2025) proposed a method using midpoint prediction instead of arbitrary intermediate states to learn a quasi-metric and geodesic generation from local information of a quasi-metric structure. This method has the advantage of preventing not only overestimation but also underestimation of distance. It is based on the assumption that infinitesimal information of quasi-metrics is known in the context of path planning, and therefore does not serve as a general solution for GRRL.

In the context of offline GRRL, Kobanda et al. (2025) proposed a method to run the Floyd-Warshall algorithm on the graph constructed from the learned quasi-metric of the latent space to improve the performance for long-horizon cases. Zheng et al. (2025) proposed a method to learn a quasi-metric with multistep TD learning, which uses intermediate waypoints on the trajectory instead of next states. Since our setting is online, we cannot directly compare their methods with ours, but their motivation that the consideration of intermediate states is effective for learning the global quasi-metric structure is similar to ours.

3 Preliminaries

We review the GRRL framework, the quasi-metric learning approach for GRRL, and midpoint learning for path planning.

3.1 Quasi-metric Space

A *quasi-metric space* is a set X equipped with a function $d : X \times X \to \mathbb{R}$ that satisfies the following axioms [1]:

1. $d(x, x) = 0$ for all $x \in X$.
2. $d(x, y) \geq 0$ for all $x, y \in X$.
3. Triangle inequality: $d(x, z) \leq d(x, y) + d(y, z)$ for all $x, y, z \in X$.

For two points $x, y \in X$, the *midpoint* between x and y is a point $z \in X$ that satisfies $d(x, z) = d(z, y) = d(x, y)/2$.

Quasi-metric learning is a technique to learn a function while satisfying quasi-metric constraints at the architecture level. More precisely, a quasi-metric learning architecture is a family of functions $d_{k,\theta} : X \times X \to \mathbb{R}$ parameterized by a size k and parameters θ that satisfies the following conditions:

[1] While pseudo-quasi-metric is a more precise terminology, we do not care about the distinguishability of points and use the term "quasi-metric" for simplicity.

1. $d_{k,\theta}(x, y)$ satisfies the conditions of a quasi-metric space.
2. For any compact quasi-metric space (X, d) and desired precision $\epsilon > 0$, when k is sufficiently large, there exist parameters θ such that

$$\sup_{x,y \in X} |d_{k,\theta}(x, y) - d(x, y)| < \epsilon.$$

While various constructions of $d_{k,\theta}$ are possible, we use Interval Quasi-metric Embedding (IQE) (Wang and Isola, 2022a) in this paper.

3.2 Goal-Reaching Reinforcement Learning

We consider the following formulation of *goal-reaching reinforcement learning* (Wang et al., 2023): The environment is a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, c)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition probability distribution, and $c : \mathcal{A} \to \mathbb{R}_{\geq 0}$ is the cost (negative reward) function. Here, $\Delta(\mathcal{S})$ denotes the set of probability distributions on $\mathcal{S}$.

We train a goal-reaching policy $\pi : \mathcal{S} \times \mathcal{S} \to \Delta(\mathcal{A})$ which takes both the current state and the goal to be reached as input and outputs the action to be taken. Let $V^\pi : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ be the function of total cost to go under the policy π. Then,

$$V^\pi(s, g) = \begin{cases} 0 & \text{if } s = g, \\ \mathbb{E}_{a \sim \pi(s,g)}[c(a) + \mathbb{E}_{s' \sim \mathcal{T}(s,a)}[V^\pi(s', g)]] & \text{otherwise.} \end{cases}$$

The objective is to minimize $V^\pi(s, g)$ for all $s \in \mathcal{S}$ and $g \in \mathcal{S}$. Note that V^π is a quasi-metric on the state space when π is optimal.

Compared to the more general framework of GCRL, this formulation has a characteristic that the action cost (negative reward) is independent of the given goal. This enables us to define a quasi-metric structure on the state space. On the other hand, because a positive reward is not given when the agent is merely approaching the goal, this sparsity of rewards makes it difficult to learn the policy with conventional reinforcement learning methods.

In spite of the formulation above, in practice, the goal is not always given as a state but as a subset of the state space. To treat such cases, we follow the approach in Appendix A of Wang et al. (2023). For each possible goal subset $g \subseteq \mathcal{S}$, we add a special goal state s_g to the state space. For each state $s \in g$, we add a special costless transition to s_g, i.e., $V^\pi(s, s_g) := 0$. When a rollout to g ends successfully at $s \in g$, we consider that such a special transition is taken. This approach has the advantage that no additional network is needed, compared to the idea of training an Aimer network (Micheli et al., 2020).

3.3 TD learning for GRRL

In Wang et al. (2023), the state is encoded into a latent space $\mathcal{L}$ and a network to learn the latent transition is trained. Thus, the state space is represented by a manifold in the latent space. The value function V is modeled with a quasi-metric learning architecture, and the policy is trained with V and the latent transition model. More precisely, we train an encoder network $E : \mathcal{S} \to \mathcal{L}$ and a latent transition network $T : \mathcal{L} \times \mathcal{A} \to \Delta(\mathcal{L})$. Abusing notation, we denote the policy and value networks that take latent vectors as input by π and V, using the same symbols as those that take states as input. A policy $\pi : \mathcal{L} \times \mathcal{L} \to \Delta(\mathcal{A})$ is trained to minimize the following expected cost:

$$\mathbb{E}_{a \sim \pi(E(s), E(g))}[c(a) + \mathbb{E}_{\tilde{s}' \sim T(E(s), a)}[V(\tilde{s}', E(g))]]. \tag{1}$$

While V is trained to maximize the cost values in QRL, it can also be trained by TD learning with the policy using the following equation:

$$V(E(s), E(g)) = \mathbb{E}_{a \sim \pi(E(s), E(g))}[c(a) + \mathbb{E}_{\tilde{s}' \sim T(E(s), a)}[V(\tilde{s}', E(g))]]. \tag{2}$$

We treat this V-learning method as a baseline for our proposed method. Note that, while this method is similar to the actor-critic TD learning in Liu et al. (2023), the difference is that we use the estimated values V of states and a transition model to train the policy, instead of the estimated values Q of state-action pairs.

3.4 MIDPOINT LEARNING

In a context different from GRRL, Kasaura (2025) proposed a method using midpoint prediction to learn a quasi-metric and geodesic generation. Under the assumption that infinitesimal information of the true quasi-metric is known, this method trains two networks: a quasi-metric function V and a midpoint prediction network m. The midpoint network m is trained to predict the midpoint between two points with respect to V and the quasi-metric function V is trained to be coincident with the given metric locally and consistent with m globally. In other words, they are trained to satisfy the following equations[2]:

$$V(x, y) = V(x, m(x, y)) + V(m(x, y), y), \tag{3}$$

and

$$m(x, y) \in \arg\min_{z \in \mathcal{L}} \max\{V(x, z), V(z, y)\}. \tag{4}$$

This approach has the following soundness property: under mild assumptions, if V coincides with the true quasi-metric locally and m and V satisfy (3) and (4), then the learned V coincides with the true quasi-metric globally. For the precise (modified) statement and its proof, see Appendix A.

4 METHOD

The main idea of our proposed method, *Auxiliary Midpoint Learning* (AML), is as follows: In addition to the networks for the above-described baseline method, we also train a network $m : \mathcal{L} \times \mathcal{L} \to \mathcal{L}$ to predict the midpoint between two points with respect to the quasi-metric V. In addition to the standard TD learning equation (2), we also want to learn V and m such that (3) and (4) are satisfied.

As for m, in addition to locating the midpoint as described above, its outputs must lie on the image $E(\mathcal{S})$ of the encoder, i.e., they must correspond to actual states. To do this, we use the Generative Adversarial Network (GAN) framework (Goodfellow et al., 2020) and introduce another network $D : \mathcal{L} \to \mathbb{R}$ that determines whether a point in the latent space corresponds to a state.

As the basis of the learning algorithm, we use Soft Actor-Critic (SAC) (Haarnoja et al., 2018), which is a popular RL algorithm for general RL with continuous action spaces. While our policy is trained with V and T instead of Q, we inherit several features from SAC: off-policy training, in which transitions collected by any past behavior policy are stored in a replay buffer and reused for gradient updates, the use of target networks, and the entropy regularization of the policy with automatic adjustment of its coefficient α.

Figure 2 shows the overview of the proposed method. For simplicity, we ignore the encoder and consider the policy and transition model to be deterministic in the figure. See the next subsection for more details.

4.1 NETWORKS AND LOSSES

We train the following networks: encoder $E : \mathcal{S} \to \mathcal{L}$, value function $V : \mathcal{L} \times \mathcal{L} \to \mathbb{R}$, policy $\pi : \mathcal{L} \times \mathcal{L} \to \Delta(\mathcal{A})$, transition $T : \mathcal{L} \times \mathcal{A} \to \Delta(\mathcal{L})$, midpoint $m : \mathcal{L} \times \mathcal{L} \to \mathcal{L}$, and discriminator $D : \mathcal{L} \to \mathbb{R}$. We use a quasi-metric learning architecture for V so that it always satisfies the quasi-metric conditions. We also keep target networks $\bar{E}$ and $\bar{V}$, which are slowly updated copies of E and V used to compute stable bootstrap targets. For a goal g and a transition (s, a, s') sampled from the replay buffer, we denote the encoded states by $\tilde{s} = E(s)$, $\tilde{g} = E(g)$, and $\tilde{s}' = E(s')$.

We train E and V so that they are consistent not only with the standard TD loss but also with m. The loss to train V and E is defined as follows:

$$L_V := \frac{1}{2}\left(V(\tilde{s}, \tilde{g}) - V_{\mathrm{TD}}\right)^2 + \frac{c_m}{2}\left(\log(V_m + \varepsilon_{\log}) - \log(V(\tilde{s}, \tilde{g}) + \varepsilon_{\log})\right)^2 \tag{5}$$

where $c_m, \varepsilon_{\log}$ are constants and

$$V_{\mathrm{TD}} := \min\{c(a) + \bar{V}(\bar{E}(s'), \bar{E}(g)), V_{\max}\}$$

[2]We changed the equation for m from the original one in Kasaura (2025). See Appendix C.

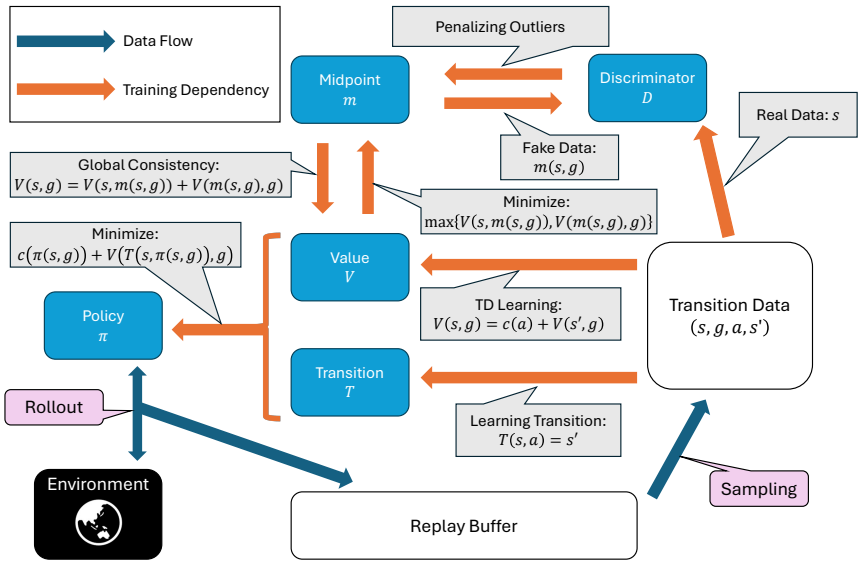


Figure 2: The (simplified) overview of the proposed method. The transition model is trained with the transition data (s, a, s'). The policy is trained with the value and transition networks. The value network is trained with both TD learning from transition data and global consistency with the midpoint network. The midpoint network is trained to predict the midpoint with respect to the value network, while outlier outputs are penalized by the discriminator network. The discriminator network is trained to distinguish real states from data and outputs of the midpoint network.

and

$$V_m := \min\{V(\tilde{s}, m(\tilde{s}, \tilde{g})) + V(m(\tilde{s}, \tilde{g}), \tilde{g}), V_{\max}\}$$

are target values. Gradients are not propagated through the target values. To prevent V from taking on large values in the early phase of training, V_{TD} and V_m are clipped with a sufficiently large constant $V_{\max}$. The loss terms are calculated differently for the two target values. While the difference from V_{TD} is local and not expected to be large due to the continuity of V, the difference from V_m tends to be significantly larger because it relates to the global consistency. Thus, the loss term for V_m is calculated with logarithms to balance the effects of the two terms. To avoid the numerical instability of the logarithm, we add a small constant $\varepsilon_{\log}$ to the arguments. While we can consider a term for the local proximity condition, we do not adopt it. See Appendix B.

The network T learns action-conditioned transitions in the latent space, and we use it to train the policy π. Specifically, from a transition datum (s, a, s'), the loss to train T is defined as

$$L_T := -\log T(\tilde{s}, a)(\tilde{s}').$$

It is effective for T to be stochastic even when the actual transition of the environment is deterministic, especially when the environment has contact-rich dynamics.

The policy loss is defined as

$$L_\pi := \mathbb{E}_{a \sim \pi(\tilde{s}, \tilde{g}),\, \tilde{s}'' \sim T(\tilde{s}, a)} \left[V(\tilde{s}'', \tilde{g}) + \alpha \log \pi(a \mid \tilde{s}, \tilde{g}) \right],$$

where the expectation is estimated with a single reparameterized sample of a and $\tilde{s}''$, and the second term is an estimate of the negative entropy of the policy. As is usual in SAC, the entropy term encourages exploration, and its coefficient α is adjusted dynamically during training so that the entropy converges to a fixed target value.

The loss to train m is defined as

$$L_m := \max\{V(\tilde{s}, m(\tilde{s}, \tilde{g})), V(m(\tilde{s}, \tilde{g}), \tilde{g})\} - \lambda_D D(m(\tilde{s}, \tilde{g})), \tag{6}$$

where the coefficient $\lambda_D > 0$ balances the geometric and adversarial terms. In actual training, we use a smooth maximum for differentiability: $\mathrm{smoothmax}(a, b) := (a + b + \sqrt{(a-b)^2 + \varepsilon_{\mathrm{sm}}})/2$, where $\varepsilon_{\mathrm{sm}}$ is a small constant.

We monitor the discriminative strength of D by the AUC-ROC score (Area Under the Receiver Operating Characteristic Curve), defined as $v_{\mathrm{AUC}} := \Pr\big[D(\tilde{s}_1) > D(m(\tilde{s}_2, \tilde{g}_2))\big]$ (with ties counted as $1/2$), where the encoded state $\tilde{s}_1$ and the pair $(\tilde{s}_2, \tilde{g}_2)$ are drawn independently from the replay buffer. In practice, it is estimated as the pairwise ranking accuracy over all pairs of encoded states and predicted midpoints in a mini-batch. Similarly to the entropy coefficient in SAC, we update λ_D to keep this score near a target value τ_{AUC}: when v_{AUC} is higher than τ_{AUC}, we increase λ_D, and when it is lower, we decrease it, so that the effect of D on the training of m remains neither too strong nor too weak.

The loss to train D is defined as

$$L_D := \mathrm{softplus}(D(m(\tilde{s}, \tilde{g}))) + \mathrm{softplus}(-D(\tilde{s})) + w_{\mathrm{gp}} J_D,$$

where $\mathrm{softplus}(x) = \log(1 + e^x)$ and w_{gp} is a fixed weight. The term J_D is a WGAN-GP-style penalty (Gulrajani et al., 2017) that keeps $\|\nabla D\|$ near a target value.

4.2 Hindsight Experience Replay with Arbitrary State Goals

We use Hindsight Experience Replay (HER) (Andrychowicz et al., 2017), which is a technique for Goal-Conditioned Reinforcement Learning with sparse rewards, including GRRL. At the beginning of training, it is difficult to collect successful rollouts to the commanded goal. To overcome this problem, HER relabels stored transitions with goals that were actually reached later in the same trajectory, and recomputes the sparse rewards accordingly. In our implementation, the relabeled goal is the final state of the episode. Note that the precise meaning of relabeling is different from the original HER in our method. In the original HER, when the goal does not specify all dimensions of the state space, the relabeled goal consists of the specified dimensions of the reached state. On the other hand, since any state can be considered as a goal even if it is not a special state for goals in our method, the relabeled goal is the reached state itself. The original HER has the following issue: even if the policy is optimal, the trajectory may not be optimal as a path to the relabeled goal. In our method, this issue does not occur. See Appendix D for the details.

5 Experiments

We compared our method (**Ours**) with the baseline method and other related methods in the benchmark environments.

5.1 Compared Methods

We compared our method with the following methods.

- **Baseline**: The quasi-metric TD learning method described in Section 3.3. This corresponds to the proposed method with $c_m = 0$ in (5).
- **QRL**: The method proposed in Wang et al. (2023), which directly maximizes the quasi-metric distances under the constraint of local proximity. Unlike the original implementation, we used a stochastic transition model. This is because we observed that a deterministic transition model is unsuitable for environments with non-smooth dynamics and resulted in extremely poor performance.
- **SAC+HER**: The standard Soft Actor-Critic method with the Hindsight Experience Replay technique. It interprets a GRRL task in a standard RL framework: an observation is a pair of a state and a goal, and a reward is given only when the state reaches the goal. It uses a value function Q for state-action-goal triples to train the policy, instead of the value function V for state-goal pairs and the transition model used in the proposed and baseline methods. The Q function is simply modeled as an MLP.
- **CRL**: The method proposed in Eysenbach et al. (2022), which uses contrastive learning so that the state-action pairs in a trajectory are close to the future state of the trajectory.

5.2 Implementation

We used the framework of JaxGCRL (Bortkiewicz et al., 2025) for the experiments, which contains the implementation of SAC+HER and CRL. We implemented our proposed and baseline methods based on the SAC+HER implementation. We also re-implemented the authors' implementation of QRL in this framework. As a quasi-metric learning method for V, we adopt IQE-sum (Wang and Isola, 2022a). While IQE-maxmean has stronger theoretical guarantees, practically IQE-sum had better performance in our experiments.

Networks are implemented as MLPs with four hidden layers and 1024 units in each layer, which is the same scale as the ones used in Appendix A.4 of Bortkiewicz et al. (2025); this is larger than those used in the main experiments of that paper.

For our method, the baseline method, and QRL, when encoding the states, we add one additional dimension to distinguish the special goal state s_g from the other states. The value of the additional dimension is 1 for s_g and 0 for other states. For all environments in these experiments, each goal specifies values for some dimensions of the state space, allowing for tolerance. For example, in maze environments, the set of goal states is the subset of states whose positions are within a certain range from the goal position. When encoding the goal states, dimensions not specified by the goal are filled with 0.0.

For details of the implementation and hyperparameters, see Appendix E.

5.3 Environments

We used four original environments in JaxGCRL (Bortkiewicz et al., 2025): **Reacher**, **Pusher Hard**, **Ant U-Maze**, and **Ant Soccer**. In addition to these environments, we added a random start (RS) option to the maze environments that randomizes not only the goal but also the starting position. This is because we believe the advantages of quasi-metric learning are better realized with this configuration. We experimented in two original maze environments with this option: **Ant U-Maze (RS)** and **Ant Big Maze (RS)**.

The total number of timesteps for training was set to 10^7 in **Reacher**, 2×10^8 in **Ant U-Maze**, and 10^8 in all other environments. We ran each method with ten seeds in **Pusher Hard** and **Ant U-Maze**, and five seeds in all other environments. See Appendix F for the details of the environments.

5.4 Results and Discussion

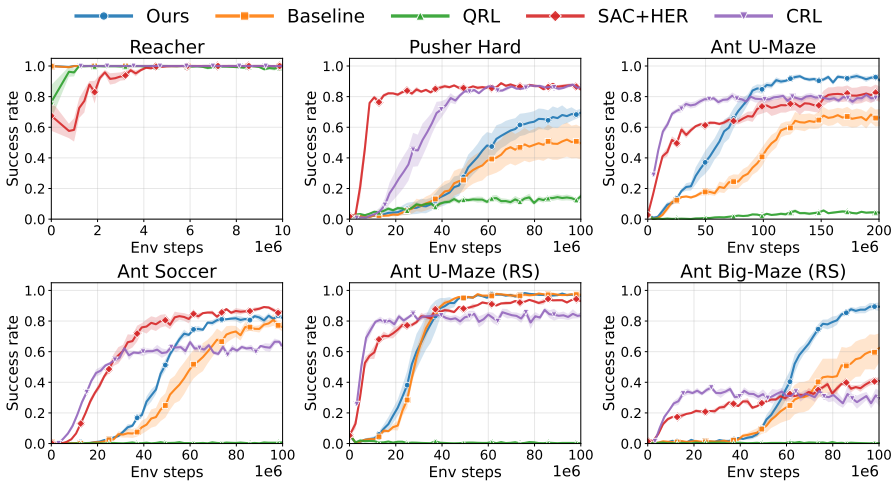


Figure 3: Averaged success rate over seeds in six environments and five methods. The shaded region shows standard errors.

Figure 3 shows the averaged success rate over seeds. Our method outperformed or matched the baseline method and QRL in all environments. Moreover, it achieved a higher success rate than any

other compared method in the Ant U-Maze and Ant Big Maze (RS) environments. The proposed method demonstrated a more pronounced advantage in the larger maze (Ant Big Maze (RS)) than in the smaller one (Ant U-Maze (RS)).

On the other hand, in the Pusher Hard and Ant Soccer environments, the proposed method underperformed compared to other methods such as SAC+HER or CRL, and even in environments where it outperformed them, success rates initially increased slowly. This may be because learning the quasi-metric structure of the whole state space is heavy and not always necessary for the success of goal-reaching. For example, in the Pusher Hard environment, the successful trajectories do not vary much depending on the position of the goal, because the arm must push the object in any case. Therefore, it is probably more efficient to learn only how to reach the goal.

While CRL showed a rapid increase in success rate, the final success rate tended to be lower. This may be explained by the limited representation ability of contrastive learning, which can find an approximate representation but not a precise one. SAC+HER showed stable performance in all environments, but its success rate increased slowly, especially in the larger maze. Consideration of the quasi-metric structure appears more important in longer-horizon tasks. Note that, since the numbers of networks used differ between methods, comparison with the same network size is not precisely fair: the quasi-metric methods (Ours, Baseline, and QRL) use an encoder before the value function and are thus effectively deeper than SAC+HER and CRL. However, the enlargement of the network size did not improve the performance of SAC+HER and CRL. See Appendix G.

QRL had poor performance in all environments except Reacher. Since we re-implemented QRL in the JaxGCRL framework, we cannot rule out the possibility that it was not sufficiently tuned. On the other hand, because these environments are more difficult than those in the original QRL paper, these results do not contradict the results reported in that paper. In addition to the instability of the constrained optimization used in QRL, this may be because enforcing local conditions does not always lead to effective learning. See also Appendix B.

6 Conclusion and Future Work

We proposed Auxiliary Midpoint Learning, a new technique to improve the quasi-metric approach for goal-reaching reinforcement learning. We showed that our method actually outperforms the baseline method in the experiments.

To observe the effect of our method in simple algorithmic settings, we deliberately did not adopt any other techniques, with the exception of HER. For example:

- Curriculum learning seems important to achieve better performance in long-horizon GRRL tasks.
- While our experiments are completely online, combining our method with offline RL may be effective because learning without any prepared data may be too difficult in complex environments.
- Network architecture is simple in our experiments, but it may be effective to use more complex architectures to achieve better performance. In particular, since midpoint prediction can be discontinuous in the latent space, a diffusion policy, which can generate multimodal distributions of states, may be effective.

We also do not consider the partial observability of states. In real-world tasks, it is possible that the agent can only observe a part of the state space. For example, in manipulation tasks, the state of an object is only observable through a camera. Moreover, in maze environments, the situation in which the shape of the maze is not known in advance can be formulated as a partially observable setting by including the maze shape as a part of the state. Both practically and theoretically, the study of partially observable environments in the GRRL framework is left as future work.

Although the experimental tasks are simple in this paper, we believe that our method can be effective for learning the quasi-metric structure of the *so-called* world model, which literally models the entire world.

AI USE STATEMENT

In this work, we used generative AI tools to help write the paper and the code for the experiments. We also used generative AI tools for surveying the related work. We have not used generative AI tools for suggesting the ideas or developing algorithms.

REPRODUCIBILITY STATEMENT

We attach all code for the experiments in the supplementary material. The environment to run the experiments is specified by the files in the supplementary material. We also specify the seeds used for the experiments. However, due to the nature of JAX, the behavior may not be fully reproduced even if the same seed is used. The proof of the claimed theorem is included in the appendix.

REFERENCES

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Michał Bortkiewicz, Władysław Pałucki, Vivek Myers, Tadeusz Dziarmaga, Tomasz Arczewski, Łukasz Kuciński, and Benjamin Eysenbach. Accelerating goal-conditioned reinforcement learning algorithms and research. In *International Conference on Learning Representations*, volume 2025, pages 15732–15754, 2025.

Vikas Dhiman, Shurjo Banerjee, Jeffrey M Siskind, and Jason J Corso. Floyd-warshall reinforcement learning: Learning from past experiences to reach new goals. *arXiv preprint arXiv:1809.09318*, 2018.

Yuanlin Duan, Wensen Mao, and He Zhu. Learning world models for unconstrained goal navigation. *Advances in Neural Information Processing Systems*, 37:59236–59265, 2024.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 35:35603–35620, 2022.

Robert W Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

Tom Jurgenson, Or Avner, Edward Groshev, and Aviv Tamar. Sub-goal trees a framework for goal-based reinforcement learning. In *International Conference on Machine Learning*, pages 5020–5030. PMLR, 2020.

Leslie Pack Kaelbling. Learning to achieve goals. In *IJCAI*, volume 2, pages 1094–8. Citeseer, 1993.

Kazumi Kasaura. Generation of geodesics with actor-critic reinforcement learning to predict midpoints. *Journal of Machine Learning Research*, 26(212):1–36, 2025.

Anthony Kobanda, Waris Radji, Mathieu Petitbois, Odalric-Ambrym Maillard, and Rémy Portelas. Offline goal-conditioned reinforcement learning with projective quasimetric planning. *arXiv preprint arXiv:2506.18847*, 2025.

Bo Liu, Yihao Feng, Qiang Liu, and Peter Stone. Metric residual network for sample efficient goal-conditioned reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8799–8806, 2023.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in neural information processing systems*, 34:24379–24391, 2021.

Vincent Micheli, Karthigan Sinnathamby, and François Fleuret. Multi-task reinforcement learning with a planning quasi-metric. *arXiv preprint arXiv:2002.03240*, 2020.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320. PMLR, 2015.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Tongzhou Wang and Phillip Isola. Improved representation of asymmetrical distances with interval quasimetric embeddings. *arXiv preprint arXiv:2211.15120*, 2022a.

Tongzhou Wang and Phillip Isola. On the learning and learnablity of quasimetrics. *arXiv preprint arXiv:2206.15478*, 2022b.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, pages 36411–36430. PMLR, 2023.

Bill Chunyuan Zheng, Vivek Myers, Benjamin Eysenbach, and Sergey Levine. Multistep quasimetric learning for scalable goal-conditioned reinforcement learning. *arXiv preprint arXiv:2511.07730*, 2025.

A Soundness Theorem of Midpoint Learning

The exact statement of the soundness of midpoint learning and its proof are as follows. Since we change the midpoint equation from the original one in Kasaura (2025) (see Appendix C), the statement and its proof are slightly modified. However, the essence of the proof is the same.

Theorem 1. *Let (X, d) be a quasi-metric space such that every pair of points has a midpoint and $V : X \times X \to \mathbb{R}$ and $m : X \times X \to X$ be functions. We assume that m is uniformly continuous and that, for any $x, y \in X$,*

$$V(x, y) = V(x, m(x, y)) + V(m(x, y), y), \tag{7}$$

$$m(x, y) \in \arg\min_{z \in X} \max\{V(x, z), V(z, y)\}, \tag{8}$$

and

$$d(x, m(x, x)) = d(m(x, x), x) = 0. \tag{9}$$

Let $\varepsilon \in (0, 1/3)$. If there exists $\delta > 0$ such that V approximates d locally for pairs with distances less than δ, which means that

$$(1 - \varepsilon)d(x, x') \leq V(x, x') \leq (1 + \varepsilon)d(x, x') \text{ for any } x, x' \in X \text{ with } d(x, x') < \delta, \tag{10}$$

then V approximates d globally, which means that

$$(1 - \varepsilon)d(x, y) \leq V(x, y) \leq (1 + \varepsilon)d(x, y) \text{ for any } x, y \in X. \tag{11}$$

In particular, if such $\delta > 0$ exists for any $\varepsilon > 0$, then $V = d$ and $m(x, y)$ is a midpoint between x and y for any $x, y \in X$.

Note that the condition (9) is always satisfied when $m(x, x) = x$.

Proof. The proof is essentially the same as the proof of Proposition 2 in Kasaura (2025), while some details are modified.

We assume that $\delta > 0$ satisfies the condition for $\varepsilon \in (0, 1/3)$.

We first prove that $V(x, y) \leq (1 + \varepsilon)d(x, y)$ for any $x, y \in X$.

We prove that $V(x, y) \leq (1 + \varepsilon)d(x, y)$ when $d(x, y) < 2^n\delta$ by induction for $n \in \mathbb{N}$. For the case $n = 0$, this is a part of (10). We assume the induction hypothesis is true for n and take $x, y \in X$ such that $d(x, y) < 2^{n+1}\delta$. Let M be a midpoint between x and y. Then,

$$\begin{aligned} V(x, y) &= V(x, m(x, y)) + V(m(x, y), y) \\ &\leq 2 \max\{V(x, m(x, y)), V(m(x, y), y)\} \\ &\leq 2 \max\{V(x, M), V(M, y)\} \\ &\leq 2(1 + \varepsilon) \max\{d(x, M), d(M, y)\} \\ &= (1 + \varepsilon)d(x, y) \end{aligned} \tag{12}$$

where the first equality comes from (7), the second inequality comes from (8), and the third inequality comes from the induction hypothesis and $d(x, M) = d(M, y) < 2^n\delta$. Thus, $V(x, y) \leq (1 + \varepsilon)d(x, y)$. By induction, for all $x, y \in X$,

$$V(x, y) \leq (1 + \varepsilon)d(x, y). \tag{13}$$

Next, we prove that $(1 - \varepsilon)d(x, y) \leq V(x, y)$ for any $x, y \in X$. From the assumption of uniform continuity of m, we can take $\delta' > 0$ such that, for any $x, y, z \in X$ with $d(y, z) < \delta'$,

$$\max\{d(m(x, y), m(x, z)), d(m(y, x), m(z, x))\} < \delta. \tag{14}$$

In particular, by considering the cases where (x, y, z) is (x, x, y) or (y, x, y), when $d(x, y) < \delta'$,

$$\max\{d(m(x, x), m(x, y)), d(m(x, y), m(y, y))\} < \delta. \tag{15}$$

From (9) and the triangle inequality,

$$\max\{d(x, m(x, y)), d(m(x, y), y)\} < \delta. \tag{16}$$

Since $\varepsilon < 1/3$, we have $\frac{1+\varepsilon}{2(1-\varepsilon)} < 1$. Using the uniform continuity of m again, we can take $\eta' > 0$ such that, for any $x, y, z \in X$ with $d(y, z) < \eta'$,

$$\max\{d(m(x, y), m(x, z)), d(m(y, x), m(z, x))\} < \left(1 - \frac{1 + \varepsilon}{2(1 - \varepsilon)}\right)\delta'. \tag{17}$$

Let $\eta := \min\{\eta'/2, \delta'\}$.

We prove by induction for $n \in \mathbb{N}$ that, when $d(x, y) < \delta' + n\eta$,

$$(1 - \varepsilon)d(x, y) \leq V(x, y), \tag{18}$$

$$(1 - \varepsilon)d(x, m(x, y)) \leq V(x, m(x, y)), \tag{19}$$

$$(1 - \varepsilon)d(m(x, y), y) \leq V(m(x, y), y). \tag{20}$$

Note that (18) follows from (19) and (20) because

$$\begin{aligned} (1 - \varepsilon)d(x, y) &\leq (1 - \varepsilon)(d(x, m(x, y)) + d(m(x, y), y)) \\ &\leq V(x, m(x, y)) + V(m(x, y), y) \\ &= V(x, y). \end{aligned} \tag{21}$$

For the case $n = 0$, if $d(x, y) < \delta'$, by the condition of δ', $d(x, m(x, y))$ and $d(m(x, y), y)$ are smaller than δ. Thus, by the condition of δ, (19) and (20) hold.

We assume the induction hypothesis is true for n and take x, y such that $d(x, y) < \delta' + (n + 1)\eta$. If $d(x, y) < \delta' + n\eta$, the conclusions hold by the induction hypothesis. Thus, we can assume that $\delta' + n\eta \leq d(x, y)$. By taking midpoints recursively, we can take $z \in X$ such that $d(x, z) + d(z, y) =$

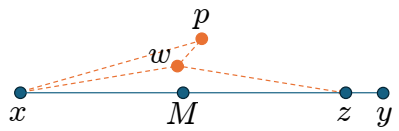


Figure 4: Positions of x, y, z, w, M, p.

$d(x, y)$ and $\eta \leq d(z, y) < 2\eta$ as shown in Figure 4. Let $w := m(x, z)$ and $l := d(x, z)$. Let M be a midpoint between x and z. Then, by (8) and (13),

$$V(x, w) \leq \max\{V(x, w), V(w, z)\} \leq \max\{V(x, M), V(M, z)\} \leq \frac{1+\varepsilon}{2} l. \tag{22}$$

Since

$$l = d(x, y) - d(z, y) < \delta' + n\eta,$$

by (19) in the induction hypothesis,

$$d(x, w) \leq (1-\varepsilon)^{-1} V(x, w) \leq \frac{1+\varepsilon}{2(1-\varepsilon)} l.$$

Let $p := m(x, y)$. Then,

$$d(x, p) \leq d(x, w) + d(w, p) < \frac{1+\varepsilon}{2(1-\varepsilon)} l + \left(1 - \frac{1+\varepsilon}{2(1-\varepsilon)}\right) \delta' < \delta' + n\eta, \tag{23}$$

where the second inequality comes from $d(z, y) < 2\eta$ and the way η is given. Thus, by (18) in the induction hypothesis, $(1-\varepsilon)d(x, p) \leq V(x, p)$. By a symmetric argument, we can also prove $(1-\varepsilon)d(p, y) \leq V(p, y)$.

Therefore, $(1-\varepsilon)d(x, y) \leq V(x, y)$ for any $x, y \in X$.

Thus, if the assumption holds for any $\varepsilon > 0$, $V = d$. Then, by (8) and the assumption that there exists a midpoint between any two points, $m(x, y)$ is a midpoint between x and y for any $x, y \in X$. □

B Discussion on Local Proximity

Since transition data tell us that consecutive states are close, it sounds reasonable to add a local proximity term to the loss function. More specifically, we can consider the following loss function instead of L_V in (5):

$$L'_V := L_V + \frac{c_{\text{local}}}{2} (V(\tilde{s}, \tilde{s}') - c(a))^2,$$

where c_{local} is a constant.

However, this loss function was not effective for learning the quasi-metric structure in our experiments. The hypothetical explanation we have is that strict satisfaction of the local condition can lead to a collapse of the latent space and, in successful learning, local distances of neighboring states are sometimes exaggerated to help the training of the policy. Despite the theoretical result in Appendix A, practically, the role of V is not to precisely approximate the true value function but to help the learning of the policy. This may also be the reason why QRL, which constrains local proximity for the quasi-metric structure, failed to learn a successful policy in some environments.

C Discussion on Midpoint Equation

In Kasaura (2025), the equation for m is

$$m(x, y) \in \operatorname*{arg\,min}_{z \in \mathcal{L}} \left(V(x, z)^2 + V(z, y)^2\right)$$

instead of (4).

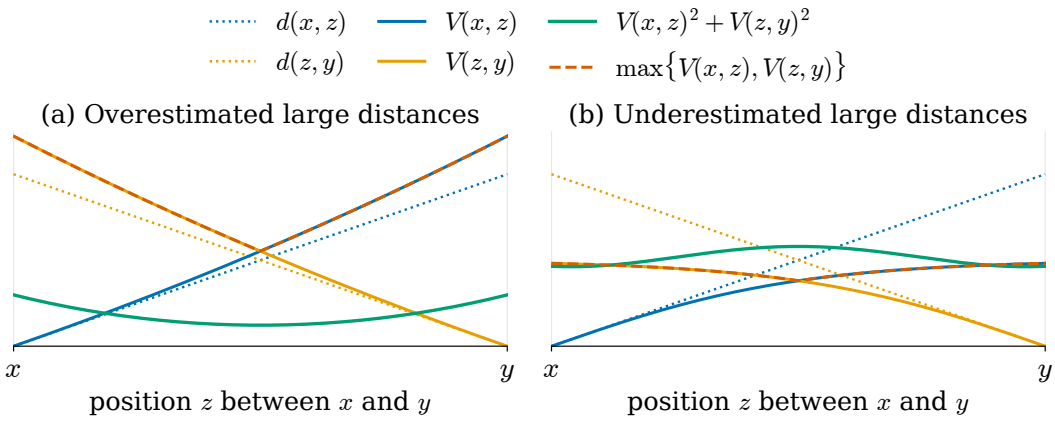


Figure 5: Graphs of the sum of the squares and the maximum value between two points for different distance estimation biases. The left figure shows the case where the larger distance tends to be overestimated, and the right figure shows the case where the larger distance tends to be underestimated.

This difference stems from the fact that Kasaura (2025) do not use a quasi-metric learning architecture for V, whereas we do. When V is trained as an arbitrary function, the distance between distant states tends to be overestimated because the successful trajectories between them are rarely found. On the other hand, when V is trained with a quasi-metric learning architecture, the distance between distant states tends to be underestimated because of the triangle inequality. As shown in Figure 5, when larger distances are overestimated, the sum of the squares is minimized at a middle point. However, when larger distances are underestimated, the sum of the squares is minimized at the endpoints. In both cases, the maximum value is minimized at the middle point while it is not smooth at this point.

D Discussion on Hindsight Experience Replay

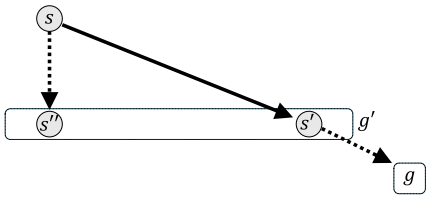


Figure 6: An example of a relabeled trajectory that can become unreasonable. The trajectory from s to s' is reasonable as a partial path to achieve the goal g. However, if g' is given as a goal, it is more reasonable to choose the trajectory to s'' instead of s' to achieve the goal g'.

We consider the case shown in Figure 6. Let s be a state and g be a goal. Let s' be a state that is reached from s by the policy when the goal is g and let g' be a goal including s'. Then, the original HER relabels the trajectory from s to s' under the condition that the goal is g as the trajectory from s to s' under the condition that the goal is g'. Then, even if the trajectory is reasonable as a partial path to achieve the goal g, it may not be reasonable as a path to achieve the goal g' because there may exist $s'' \in g'$ such that the distance from s to s'' is short and the distance from s'' to g is not short. In such a case, if the given goal is actually g' instead of g, the reasonable policy will choose the trajectory to s'' instead of s' to achieve the goal g'.

This example shows that the relabeled trajectory can be a source of noise for learning the optimal policy. On the other hand, if the objective is to increase the success rate, it might not be problematic. Furthermore, since relabeling will occur less frequently as the success rate increases, even if optimality is the objective, it may not matter in the end.

In any case, in the framework of our method, since any state can be considered as a goal, s' itself can be considered as a goal. Therefore, this issue does not occur: if the trajectory from s to s' is reasonable as a partial path to achieve the goal g, it is also reasonable as a path to reach s'.

E Implementation Details

All experiments were run with the JaxGCRL framework (Bortkiewicz et al., 2025). Our method (AML) and the Baseline share the same implementation; Baseline disables midpoint learning, which corresponds to setting $c_m = 0$ in (5). QRL is a re-implementation in the same framework; SAC+HER and CRL use the JaxGCRL implementations.

For SAC+HER and CRL, we use the same hyperparameters as the experiments with the larger network size (Appendix A.4) in Bortkiewicz et al. (2025). The only exception is the number of environments in the SAC+HER setup, which is 512 (the default value in the JaxGCRL repository) instead of 1024 (the value used in the paper), but this does not appear to have affected the results.

The implementation of QRL was generally faithful to the original implementation in the repository of Wang et al. (2023), but the transition model was made stochastic to improve performance. While the hyperparameters were also set according to the original implementation, the network settings were set to be the same as those of the other methods.

For AML and Baseline, we tuned only the learning rate in the range from 10^{-6} to 10^{-4} and the Polyak averaging rate τ in the range from 0.001 to 0.005. Other hyperparameters were not investigated in the experiments of this paper.

Each run was executed on a single NVIDIA H200 GPU.

Networks. Every MLP has four hidden layers of width 1024 with ReLU activations (swish activations for CRL). For AML and Baseline, the encoder maps an augmented state (state plus a binary flag that is 1 only for the special goal state s_g) to a latent of dimension 1024, with LayerNorm on the encoder. Unspecified goal dimensions are filled with 0. The quasi-metric V is an IQE-sum model (Wang and Isola, 2022a) with embedding shape $(64, 32)$. The policy is a tanh-squashed diagonal Gaussian and the latent transition is a diagonal Gaussian, both conditioned on latents (transition log-standard deviations are clipped to $[-5, 2]$). QRL uses the same MLP width but a latent dimension of 128, two critic heads, and LayerNorm on all networks. SAC+HER uses LayerNorm MLPs for the policy and Q; CRL uses LayerNorm MLPs with representation dimension 64, symmetric InfoNCE, and a norm energy function.

Optimization and data. We use Adam. For AML/Baseline, all network optimizers apply global gradient clipping with threshold 1.0. Training uses 512 parallel environments (1024 for CRL), episode length 1001, batch size 256, unroll length 62, replay capacity 10^4, and minimum replay size 10^3. Evaluation uses 256 parallel environments and 50 evaluations over training. The target networks $\bar{E}$ and $\bar{V}$ (or those of Q for SAC) are updated by Polyak averaging with rate τ. For

Table 1: Shared training settings (all methods unless noted).

Hyperparameter	Value
Hidden layers $\times$ width	4×1024
Batch size	256
Unroll length	62
Replay buffer size	10^4
Min. replay size	10^3
Episode length	1001
Parallel envs (train)	512 (1024 for CRL)
Parallel envs (eval) / #evals	256 / 50
Entropy coef. α LR	3×10^{-4}
Entropy coef. α init	1
Target entropy	$-0.5 \cdot \lvert\mathcal{A}\rvert$

Table 2: Method-specific hyperparameters. For QRL, the value/critic LR applies to the value, encoder, and transition networks. The discount factor γ is the TD discount for SAC+HER and the future-goal sampling weight for CRL. Entries marked "–" are unused.

	Ours	Baseline	QRL	SAC+HER	CRL
Value/critic LR	10^{-5}	10^{-5}	10^{-4}	10^{-4}	3×10^{-4}
Policy LR	10^{-5}	10^{-5}	3×10^{-5}	10^{-4}	6×10^{-4}
Target τ	0.002	0.002	0.005	0.005	–
Discount γ	–	–	–	0.99	0.99
LayerNorm	encoder	encoder	all	all	all
Latent/repr. dim	1024	1024	128	–	64
IQE embedding shape	$(64, 32)$	$(64, 32)$	$(64, 32)$	–	–
#critics	–	–	2	–	–
Lagrange LR	–	–	10^{-2}	–	–
CRL loss	–	–	–	–	sym-InfoNCE
CRL energy	–	–	–	–	norm
Logsumexp pen.	–	–	–	–	0.1
c_m	1	0	–	–	–
$\varepsilon_{\log}$	1	–	–	–	–
$V_{\max}$	$1.5\times$ ep. len.	$1.5\times$ ep. len.	–	–	–
$\varepsilon_{\mathrm{sm}}$	10^{-6}	–	–	–	–
λ_D init	0.01	–	–	–	–
λ_D LR	3×10^{-4}	–	–	–	–
τ_{AUC}	0.8	–	–	–	–
w_{gp}	0.1	–	–	–	–
τ_{gp}	2	–	–	–	–
Discriminator LR	10^{-6}	–	–	–	–
Global grad clip	1	1	–	–	–

AML/Baseline, $V_{\max}$ is set to 1.5 times the episode length, since the true values of V for reachable pairs are bounded by the episode length. The entropy coefficient α is parameterized as $\exp(\log\alpha)$ with initial value 1 and is updated to keep the policy entropy near the target $\bar{H} = -0.5 \cdot |\mathcal{A}|$. The loss function of $\log\alpha$ is defined as

$$L_{\log\alpha} := \alpha\big(-\log\pi(a \mid \tilde{s}, \tilde{g}) - \bar{H}\big),$$

where $a \sim \pi(\tilde{s}, \tilde{g})$.

AML-specific losses. The midpoint consistency term in L_V uses the log offset $\varepsilon_{\log}$ in (5) with $c_m = 1$ and $\varepsilon_{\log} = 1$. The nonsmooth $\max$ in L_m is replaced by $\mathrm{smoothmax}$ with $\varepsilon_{\mathrm{sm}} = 10^{-6}$. The discriminator learning rate is 10^{-6}. The adversarial weight λ_D is parameterized as $\exp(\log\lambda_D)$ with initial value 0.01 and is updated to keep the discriminator AUC near $\tau_{\mathrm{AUC}} = 0.8$. The loss function of $\log\lambda_D$ is defined as

$$L_{\log\lambda_D} := \log\lambda_D\,(\tau_{\mathrm{AUC}} - v_{\mathrm{AUC}})\,.$$

To avoid numerical overflow, $\log\lambda_D$ is clipped from above at 70 after each update. The WGAN-GP-style penalty uses fixed weight $w_{\mathrm{gp}} = 0.1$ and target $\tau_{\mathrm{gp}} = 2$. To calculate the penalty, we sample $u \sim \mathrm{Unif}(0, 1)$, take the interpolated latent $\tilde{z} = u\tilde{s} + (1-u)m(\tilde{s}, \tilde{g})$, and set

$$J_D := \mathbb{E}\Big[\big(\|\nabla_{\tilde{z}} D(\tilde{z})\|_2 - \tau_{\mathrm{gp}}\big)^2\Big]\,.$$

Hyperparameters. Tables 1 and 2 list the hyperparameters used in the reported runs. Environment-specific training horizons and seed counts are as in the main text (Section 5).

F Environment Details

We use six continuous-control goal-reaching tasks. Four are taken from JaxGCRL (Bortkiewicz et al., 2025) (Reacher, Pusher Hard, Ant U-Maze, and Ant Soccer), and the remaining two are Ant

maze variants with randomized starts. In all tasks, the reward is sparse (success when the relevant distance falls below a threshold). Table 3 lists the state, goal, and action dimensions.

Reacher. A 2D two-link arm must place its tip at a target sampled uniformly from a disk around the origin (success threshold 0.05).

Pusher Hard. A 3D robotic arm must push a movable object to a randomly sampled goal position on the plane (success threshold 0.1). Both the object and the goal are randomized at reset.

Ant U-Maze / Ant U-Maze (RS) / Ant Big Maze (RS). A quadruped Ant navigates a maze to an (x, y) goal chosen from maze-specific candidates (success threshold 0.5). Without RS, the start is fixed and only the goal is randomized; with RS, the start is also sampled from admissible cells. Ant Big Maze (RS) uses a larger layout than U-Maze. The layouts are shown in Figure 7 (a)–(b).

Ant Soccer. The Ant must push a spherical ball to a goal sampled on a circle of radius 5 around the origin (success threshold 0.5). The ball is initialized between the agent and the goal with small noise.

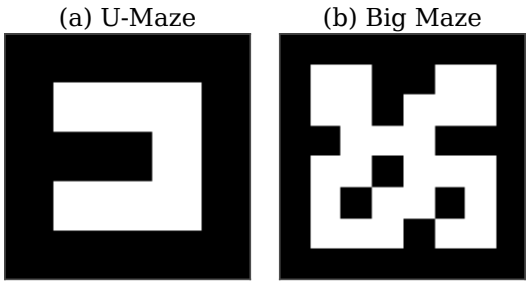


Figure 7: Maze layouts used in our experiments (black: wall, white: free space).

Table 3: State, goal, and action dimensions of the evaluation environments. The observation concatenates the state and the goal.

Environment	State dim.	Goal dim.	Action dim.
Reacher	10	3	2
Pusher Hard	20	3	7
Ant U-Maze	29	2	8
Ant Soccer	31	2	8
Ant U-Maze (RS)	29	2	8
Ant Big Maze (RS)	29	2	8

G NETWORK SIZE EXPERIMENT

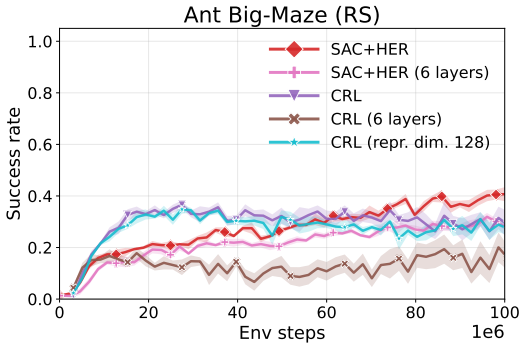


Figure 8: Averaged success rate in Ant Big Maze (RS) for methods with increased numbers of layers or an increased representation dimension. The shaded area shows the standard errors.

Since the numbers of networks differ between methods, it is difficult to compare them fairly. In particular, since the quasi-metric learning methods use encoder networks before applying the value function, they can be regarded as deeper architectures than SAC+HER and CRL. Thus, to verify that four layers are enough for SAC+HER and CRL, we ran SAC+HER and CRL with extended numbers of layers in Ant Big Maze (RS). Additionally, we ran CRL with a doubled representation dimension (128). All runs were executed with five seeds. The results are shown in Figure 8. As shown in the figure, increasing the number of layers or the representation dimension does not improve the performance.